EXHIBIT 2

Filed Under Seal

CLEMENT SETH ROBERTS (STATE BAR NO. 209203)
croberts@orrick.com
BAS DE BLANK (STATE BAR NO. 191487)
basdeblank@orrick.com
ALYSSA CARIDIS (STATE BAR NO. 260103)
acaridis@orrick.com
EVAN D. BREWER (STATE BAR NO. 304411)
ebrewer@orrick.com
ORRICK, HERRINGTON & SUTCLIFFE LLP
The Orrick Building
405 Howard Street
San Francisco, CA 94105-2669
Telephone: +1 415 773 5700
Facsimile: +1 415 773 5759

SEAN M. SULLIVAN (*pro hac* vice)
sullivan@ls3ip.com
MICHAEL P. BOYEA (*pro hac* vice)
boyea@ls3ip.com
COLE B. RICHTER (*pro hac* vice)
richter@ls3ip.com
LEE SULLIVAN SHEA & SMITH LLP
656 W Randolph St., Floor 5W
Chicago, IL 60661
Telephone: +1 312 754 0002
Facsimile: +1 312 754 0003

Attorneys for Sonos, Inc.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

GOOGLE LLC,

Plaintiff and Counter-defendant,

v.

SONOS, INC.,

Defendant and Counter-claimant.

Case No. 3:20-cv-06754-WHA
Related to Case No. 3:21-cv-07559-WHA

REBUTTAL EXPERT REPORT OF DOUGLAS C. SCHMIDT

"playback queue" as "a list of multimedia content selected for playback" in the context of claim 13 of the '615 Patent. *See* 20-cv-6754, Dkt. 316 [Order Granting Motion for Partial Summary Judgment as to '615 Patent].

102. In reaching its construction, I understand that the Court reasoned (i) "a list of one is still a list," (ii) "nothing requires a 'playback queue' to contain plural multimedia items" and thus, "the list must contain at least one item, but not necessarily more than one," and (iii) a user need not "directly populate and manage the queue" and thus, "the list does not necessarily… require users to select content directly." *Id.*, 7-8. While the Court stated "[t]he patent repeatedly ***associates*** a queue with a 'list' or 'playlist,'" I take the Court's statement (especially the use of the word "associates") in the context of the Court's overall order to ***not*** mean that a "playlist" is ***equivalent*** to a "playback queue." *See also, e.g.*, *id.*, 7 ("[T]he specification repeatedly describes embodiments where a queue only contains a single audio track.").

103. In applying its construction for purposes of infringement, however, the Court further defined what is required to be a "playback queue." For instance, the Court started its infringement analysis by considering which queue was used by Google's accused apps running on the accused playback devices for playback – the "local" queue or the "cloud"/"remote" queue. *Id.*, 8-9. Put another way, the Court determined that the "playback queue" needed to be the queue that was used for playback of the list of multimedia content. In finding no infringement of claim 13 of the '615 Patent, the Court agreed with Google that the accused apps running on the accused playback devices "do not use a 'local playback queue.'" *Id.*, 8-10.

104. In its Order, the Court also indicated that the "playback queue" must have a complete list of all the multimedia items that are to be played back – a "subset" or "short list" of such items is not enough to be a "playback queue." *Id.*, 9 ("The groups of three items stored by the respective apps are not lists of multimedia content selected for playback."), 10 ("The passage thus distinguishes a local playback queue from the 'short list of tracks.'").

105. The Court also indicated in its Order that a queue was not a "playback queue" if it "merely provide[d] the means to *process* the list[] for playback." *Id.*, 10 (emphasis in original).

106. And finally, the Court concluded that, "[in] short," the "playback queue" is the

"queue [that] runs the show." *Id.*

107. Thus, according to the Court's Order, I understand that the claim term "playback queue" refers to a "list of multimedia content selected for playback" with the following characteristics:

- The playback queue is the list of media items that is used for playback;
- The playback queue contains the entire list of media items selected for playback;
- The playback queue is not being used merely to process the list of media items for playback; and
- The playback queue is the queue that "runs the show."

108. Based on the Court's Order and its requirements for a "playback queue," it appears that, in a system like the YTR System and Tungsten/Q System, there can only be ***one*** playback queue, which is either remote or local (but not both). I understand that Google has taken the same position. *See, e.g.*, Dkt. 343.02 [Sonos Inc.'s Opposition to Google LLC's Motion for Leave to Amend Invalidity Contentions Pursuant to Patent L.R. 3-6], 4-6; *infra* §XI.A.

2. "Multimedia Content" versus "Media Item"

109. In its Order, the Court rejected "Google's proposal to include the term 'multimedia item' in the construction" and instead construed "playback queue" in claim 13 of the '615 Patent to recite "multimedia content" because "[t]he claim uses the term 'multimedia content,' and there is no need to introduce additional ambiguity by importing a new term." Dkt. 316, 8.

110. I understand that Dr. Bhattacharjee has applied the Court's construction of the term "playback queue" provided in the context of claim 13 of the '615 Patent ***verbatim*** to the term "playback queue" provided in the context of the '033 Patent's claims.

111. As I noted in my Opening Report, however, the '033 Patent's claims do not recite the term "multimedia content" like the '615 Patent's claims do. Instead, the '033 Patent's claims recite the term "media item." For purposes of the '033 Patent, therefore, I will interpret the Court's construction of "playback queue" (provided in the context of claim 13 of the '615 Patent) as "***a list of one or more media items selected for playback***."

112. However, my opinions would remain the same under the Court's exact construction of "playback queue" provided in the context of claim 13 of the '615 Patent and applied verbatim by Dr. Bhattacharjee because a POSITA would understand that the term "multimedia content" is

Queuing Support], 27 ("Scenarios Supported 1. Jin wants to listen to her favorite band. She uses her sender app to ***queue a playlist*** into her cast for audio device … 8. Jin loads a second playlist at the bottom of the existing one."); GOOG-SONOSWDTX-00039862 [Credential Transfer Tokens (CTTs)], 62 ("When the remote user wants to ***add a playlist to the queue***, it sends the videoId of the first video it wants to play and the listId of the playlist containing that video."); GOOG-SONOSWDTX-00053409 [How the Queue Works], 09 ("If the user chooses to play a song within an album or playlist view, the whole album or ***playlist gets added to the queue*** and the selected songs starts playing. … The ***user can drag*** songs, albums or ***playlists to the queue and put them at any point in the queue***.")

130. It is therefore my opinion that a POSITA would have understood that a "playlist" is different than a "playback queue," and I disagree with Dr. Bhattacharjee's apparent interpretation that any stored "playlist" amounts to a "playback queue."

2. Dr. Bhattacharjee Misrepresents Sonos's Infringement Theories

131. Relatedly, I disagree with Dr. Bhattacharjee's repeated mischaracterization of Sonos's position with respect to Google's infringement. In particular, Dr. Bhattacharjee incorrectly opines that Sonos's position is that just any "list of recommended videos provided to a mobile device by a server in the cloud" or "list of recommended videos generated by a cloud server" amounts to the claimed "remote playback queue." *See, e.g.*, Bhatta. Op. Report, ¶¶87, 165, 218, 228, 264, 295, 298-99, 492, 508, 515.

132. Instead, as explained in my Opening Report, Google's infringement is based on Google's YouTube cloud infrastructure that provides a list of one or more media items ***selected for playback*** (i.e., the Watch Next queue) by the Sender at times or by a Receiver at other times. *See, e.g.*, Schmidt Op. Report, ¶¶125-28, 229, 241. That list can be filled with videos recommended by the YouTube service but that is not what makes the Watch Next queue a "remote playback queue." Rather, what is relevant is that the Watch Next queue, which might contain recommended videos (or songs in the YouTube Music context), contains what is ***selected for playback*** by the Sender or Receiver. In the words of the Court, the Watch Next queue "runs the show" for the Sender and Receiver.

the Lounge Server, which will, in turn, send the new SET_VIDEO message to each Leanback Screen in the session. RealVideoPlayService.java, lns. 80-124, 249-67, 269-73; RealPlaystatePushService.java, lns. 116-35, 282-85; RemotePlayerController.java, lns. 77-89.

177. As previously discussed, the connected remotes in the session have the ability to modify or edit the playlist with updates. To do so, a remote will send an update message to the Lounge Server, which will, in turn, send the updates for the playlist to the remotes in the session and the Leanback Screens in the session via an UPDATE_PLAYLIST message (also referred to as a "updatePlaylist" message). GOOG-SONOSWDTX-00040194 at 194-195; Levai Dep. Tr. (January 6, 2023), 51:6-21; RealPlaystatePushService.java, lns. 86-114, 305-53, 345-49; RemotePlayerController.java, lns., 112-19; RealLoungeSessionManager.java, lns. 307-86, 851-83. Once the updates have been processed by the Leanback Screens, the locally-stored playback queue on each Leanback Screen will maintain the full updated playlist and each Leanback Screen will continue to playback the media items from their locally-stored playback queue. Levai Dep. Tr. (January 6, 2023), 52:5-14 59:15-60:12; LeanbackModule.as, lns. 682-705, 2144-82.

178. Accordingly, in the YTR System's remote-control mode, the playback queue locally stored on the Leanback Screen: (i) comprises the list of media items that is used for playback by the Leanback Screen; (ii) contains the entire list of media items selected for playback; (iii) is not being used merely to process a list of media items maintained elsewhere for playback; and (iv) is the queue that "runs the show."

179. In other words, if there is only one playback queue in the YTR System, it is the local playback queue on the Leanback Screens. Dr. Bhattacharjee does not dispute this conclusion. *See* Bhatta. Op. Report, ¶185.

3. <u>Dr. Bhattacharjee's Mischaracterizations of the YTR System</u>

180. In his Opening Report, Dr. Bhattacharjee attempts to explain various aspects of the YTR System. However, as explained below, I disagree with many of Dr. Bhattacharjee's characterizations.

181. For instance, as Dr. Bhattacharjee acknowledges in his Opening Report, the YTR application served as a "remote control" for a larger screen. *See* Bhatta. Op. Report, ¶157; *see also,*

messaging between the Lounge Server and Leanback Screen in party mode as "mostly the same path as, yes, in non-Party Mode," and confirming that, "based on the code," in party mode "the MDx server was mimicking the same messages that should be sent from the [] remote app when [in] non-Party Mode"), 56:25-58:19 (confirming with an unqualified "yes" that "[i]n either Party Mode or non-Party Mode, there is a set playlist message sent from the MDx Server down to the Leanback screens" and that "[i]n both Party Mode and non-Party Mode, the Leanback screens always get the entire list of videos for playback"). In other words, there is no substantive difference between party mode and non-party mode except that, in party mode, the Lounge Server has to coordinate with the guests' devices in addition to the host's devices.

201. I also disagree with Dr. Bhattacharjee's opinion that YTR party mode relied on a remote playback queue or a "cloud queue" for playback. *See* Bhatta. Op. Report, ¶186. To the contrary, it is my opinion that YTR party mode used a local playback queue on the Leanback Screens just like the normal, non-party mode of YTR.

202. Indeed, the copy of the playlist stored on the Lounge Server in party mode was not used for playback. Rather, the Lounge Server used that copy of the playlist to merely "coordinate," *i.e.*, "make sure that every device had the same order of videos," with the Leanback Screens and the remote controls that were connected in the session. Levai Dep. Tr. (January 6, 2023), 56:2-16.[18]

b. Messages for Setting Up a Playback Queue on the Leanback Screens in Party Mode

203. As in non-party mode, it is important to note that there is no direct communication between the remote controls (*e.g.*, mobile devices) and the Leanback Screens (*e.g.*, TVs) in the YTR System's party mode. Bhatta. Op. Report, ¶158. All communication between the remote controls and the Leanback Screens must be facilitated through and by the Lounge Server in both party and non-party modes. *Id.*

204. It is also important to note that in party mode only "videoIDs" are maintained in the

[18] Likewise, Dr. Bhattacharjee's reference to comments in RemoteQueueManager.java (Bhatta. Op. Report, ¶¶177, 300) is unconvincing because (i) that is a file for the YTR Application source code (i.e., a YTR remote control), (ii) as I explained in my Opening Report whether something is "remote" of something else is context dependent, and (iii) here, the comment is referring to the fact that the "remote queue" that the YTR application is managing is on the Screen(s).

53:15;RealPartyModeManager.java, lns. 190-212; RealLoungSessionManager.java, lns. 307-86, 776-799, 1111-19, 1088-109, 1218-27. The host remote control may also change the value for the playlist contain in its local playback queue from "QUEUE" to "PARTY_QUEUE." RealPartyModeManager.java, lns. 190-212, 269-78. Otherwise, there is no substantive difference between a playlist tagged as "QUEUE" versus "PARTY_QUEUE."

208. The Lounge Server saves a copy of the SET_PARTY_PLAYLIST message. Levai Dep. Tr. (January 6, 2023), 52:15-53:15. The Lounge Server then sends a SET_PLAYLIST message with the playlist from the SET_PARTY_PLAYLIST message to any and all of the connected Leanback Screens of the host and guests in the session, and sends a PARTY_PLAYLIST_MODIFIED message to all other remote controls of the host and guests connected to the session. Bhatta. Op. Report, ¶189 ("This sendPartyQueueToScreen function sends a SET_PLAYLIST [message] from the Lounge server to the connected Screens. The SET_PLAYLIST message sent from the Lounge server to the Screens contains the cloud hosted party queue."); Levai Dep. Tr. (January 6, 2023), 52:15-54:18, 57:13-58:6; RealLoungeSessionManager.java, lns. 676-763, 1111-19, 1218-27. The Lounge Server uses its saved copy of the playlist from the SET_PARTY_PLAYLIST message to relay to any new Leanback Screens and/or remote controls that join the existing party mode session a copy of the playlist. Levai Dep. Tr. (January 6, 2023) at 52:15-53:15, 56:6-16; RealLoungeSessionManager.java, lns., 237-69, 1065-72, 1074-86, 1088-109.

209. The SET_PLAYLIST message that each Leanback Screen receives from the Lounge Server in party mode is the same as the SET_PLAYLIST message that each Leanback Screen receives from the Lounge Server in non-party mode. Bhatta. Op. Report, ¶193 ("As I discussed above, transferring playback from a mobile device to a Screen causes the Lounge server to send a SET_PLAYLIST message and/or SET_VIDEO message from the Lounge server to the Screens to transfer playback. In particular, when not in party mode a SET_PLAYLIST message is sent from the Lounge Server to the Screens to send the queue and start playback of the queue. In party mode, the Lounge Server sends a SET_PLAYLIST message to the Screens to send the party queue and a further SET_VIDEO message is used to then start playback of the queue.");

RealLoungeSessionManager.java, lns., 307-86, 851-83, 1030-42, 1088-109. After receiving this SET_PLAYLIST message from the Lounge Server, each Leanback Screen in the party mode sessions saves a copy of the entire playlist into its local playback queue. Levai Dep. Tr. (January 6, 2023), 54:19-55:10, 58:16-9; LeanbackModule.as, lns. 167, 668-77, 682-705, 1015-50; LeanbackV2Module.as, lns. 541-46. Just as in a non-party mode session, each Leanback Screen in a party mode session uses its own local playback queue, which now contains the entire party playlist, to playback the media items sent from the remote control(s). Bhatta. Op. Report, ¶189 ("The Screens receive the SET_PLAYLIST message and perform additional processing and messaging to play the videos in [their] queue . . . ."); *id.*, ¶¶193-197; Levai Dep. Tr. (January 6, 2023), 54:19-55:10, 58:16-9; LeanbackModule.as, lns. 167, 2144-82; LeanbackV2Module.as, lns. 541-46; VideoApplication.as, lns. 859-63.

210. Mr. Levai's deposition testimony confirms that a Leanback Screen in party mode uses its own copy of the party playlist stored in its local queue for playback without any dependency on the copy of the playlist on the Lounge Server:

> Q Well, all the devices such as the Leanback screens and the YouTube remotes, they also store and maintain a copy of that playlist; correct?
>
> A Well, they need to so that they can show it in [GUI] to the user.
>
> Q And so they can play it back; right?
>
> A Well, in remote mode, yes, the lounge screen plays back those videos.
>
> Q Plays back those videos stored locally on the screen; right?
>
> A Well, it gets a list of videos and which one to play and starts playing that video. And then in order to know which video to play next when the current one ends, it refers to that list.
>
> Q The locally stored list on the Leanback screens; correct?
>
> A Yes. I don't believe it asked the MDx server for which video to play next when the current ended. I believe it referred to that list that it caught before.

Levai Dep. Tr. (January 6, 2023), 59:15-60:12.

211. As long as the session is active, and absent any modifications or edits to the party playlist on a remote control, each Leanback Screen in a party mode session will continue to playback the media items in its locally-stored playback queue in the order set by that queue and

until that queue is empty. Levai Dep. Tr. (January 6, 2023), 55:19-25; LeanbackModule.as, lns. 167, 1435-50, 1308-64; VideoApplication.as, lns. 852-57.

212. Just like in non-party mode, the SET_PARTY_PLAYLIST and SET_PLAYLIST messages in party mode have no way of indicating which video in the playlist should be played, and thus, a remote control will need to send a SET_VIDEO message to the Lounge Server to indicate which video should be played on the Leanback Screen(s). Bhatta. Op. Report, ¶191 ("For instance, if a user presses the Connect/Reconnect button while in party mode, the YTR application will transfer playback to the Screens by sending a SET_VIDEO message to the Lounge server."); Levai Dep. Tr. (January 6, 2023), 54:11-18; RemotePlayerController.java, lns. 77-89. The Lounge Server, in turn, sends the SET_VIDEO message to each Leanback Screen in the party mode session. Bhatta. Op. Report, ¶191 ("The SET_VIDEO message is then relayed by the Lounge server to the Screens to transfer playback."); RealLoungeSessionManager.java, lns. 307-86, 851-83, 1030-42. To change the currently playing video, the remote control will need to send a new SET_VIDEO message to the Lounge Server, which will, in turn, send the new SET_VIDEO message to each Leanback Screen in the party mode session. RemotePlayerController.java, lns. 77-89; RealLoungeSessionManager.java, lns. 307-86, 851-83, 1030-42.

213. As previously discussed with respect to a non-party mode session, the connected remote controls in a party mode session have the ability to modify or edit the playlist with updates. To do so, a remote control will send update messages to the Lounge Server, which will, in turn, send the updates for the playlist to the remote controls in the session via a PARTY_PLAYLIST_MODIFIED message and to the Leanback Screens in the session via an UPDATE_PLAYLIST message. Levai Dep. Tr. (January 6, 2023), 55:11-18; SharedPlaylistContentService.java, lns. 30-122; RemoteQueueManager.java, lns. 16, 20-82; RealLoungeSessionManager.java, lns. 676-763, 689-760, 1121-24, 1218-27, 1229-38. Notably, the PARTY_PLAYLIST_MODIFIED message, as well as the UPDATE_PLAYLIST message, "contains the entire shared party queue." Bhatta. Op. Report, ¶175; RealLoungeSessionManager.java, lns. 1218-27, 1229-38.

214. The UPDATE_PLAYLIST message that each Leanback Screen receives from the

Lounge Server in party mode is the same as the UPDATE_PLAYLIST message that each Leanback Screen receives from the Lounge Server in non-party mode. RemotePlayerController.java lns. 112-19; RealLoungeSessionManager.java, lns. 1229-38. Once the updates have been processed by the Leanback Screens, the locally-stored playback queue on each Leanback Screen in the party mode session will maintain the full updated playlist and each Leanback Screen in the party mode session will continue to playback the media items from their locally-stored playback queue. Levai Dep. Tr. (January 6, 2023), 54:19-55:10, 58:16-9, 59:15-60:12; LeanbackModule.as, lns. 167, 682-705, 1435-50, 1308-64; VideoApplications.as, lns. 852-57.

215. Accordingly, in the YTR System's remote-control party mode, the playback queue locally stored on the Leanback Screen: (i) comprises the list of media items that is used for playback by the Leanback Screen; (ii) contains the entire list of media items selected for playback; (iii) is not being used merely to process a list of media items maintained elsewhere for playback; and (iv) is the queue that "runs the show."

216. In other words, as with non-party mode, if there is only one playback queue in the YTR System's party mode, it is the local playback queue on the Leanback Screens.

B. U.S. Patent No. 9,490,998

217. Dr. Bhattacharjee relies on U.S. Patent No. 9,490,998 (the "'998 Patent") to support his YTR obviousness theories, namely, that it would have been obvious to add a "device-picker" to the YTR System. I disagree with Dr. Bhattacharjee that the '998 Patent discloses a device-picker as explained in my Rebuttal Report for the Court's Patent Showdown Procedure, which is incorporated herein by reference. I also disagree with Dr. Bhattacharjee that it would have been obvious to combine the YTR System with the '998 Patent to render the Asserted Claims of the '033 Patent obvious. In fact, the YTR System's party mode is completely incompatible and inoperable with a device-picker.

218. The '998 Patent is entitled "Network-Based Remote Control" and issued from an application filed on March 7, 2011, which claims priority to November 8, 2010. The '998 Patent lists Google as the assignee. One of the named inventors of the '998 Patent is Ms. Bobohalma, who stated, in her declaration, that "[t]his patent discloses ***some*** of the work that I did on the

- "**THE COURT:** All right. So, Mr. Verhoeven, the argument against you here is that -- there are two queues: There is one in the Cloud, and there is another one on the speaker. The local playback queue is on the speaker, and all it needs to know is what is the next one. And so that's good enough for its purposes and just calls the next one. What do you say to that? **MR. VERHOEVEN [FOR GOOGLE]:** I say that that's unpersuasive in the extreme . . .We're talking about ***a*** queue. ***Somebody has to own that queue***. Somebody has to ***maintain that queue***. Somebody has to be in charge of ***that queue***. Who is it? It's the Cloud. That's why it's called ***the*** Cloud ***queue***. ***The queue*** is maintained in the Cloud. If you want to know -- if you're the speaker and you want to know the next item in ***the queue***, you ask the Cloud because the Cloud maintains ***the queue***." *Id.*, 59-60;

- "In the accused YouTube system, ***the playback queue*** is not stored on a playback device, such as the playback device. Rather, as already mentioned, it is stored in the Cloud Queue. *See, e.g.*, supra ¶49. Indeed, the protocol used by the accused YouTube system for controlling playback is Version 3 of the MDx protocol, which was ***changed from earlier version of MDx***—such as ***Version 1 that is used in the prior art YT Remote*** discussed in Section VII—to ***eliminate the "local playback queue" in favor of a Cloud Queue***." Bhatta. Showdown Decl., ¶64;

- "The GPM system does not infringe because it uses a Cloud Queue, not a 'local playback queue on the particular playback device.'" *Id.*, ¶113; *see also*, *id.*, ¶¶115-20;

304. I understand that the Court sided with Google based on Google and Dr. Bhattacharjee's representations and ordered summary judgment that the YTR System invalidated claim 13 of the '615 Patent but that the accused YouTube apps did not infringe claim 13 of the '615 Patent. 20-cv-6754, Dkt. 316 [Order Granting Motion for Partial Summary Judgment as to '615 Patent], 10. In this regard, I understand that the Court accepted Google and Dr. Bhattacharjee's representations that a system cannot have both a "local playback queue" and a "remote playback queue"/"cloud queue" because "locally-stored information is merely a mirror reflecting a subset of what is happening in the cloud queue." *Id.*, 9-10. I understand that, according to the Court, groups of three items (such as that stored by a playback device when used with the accused YouTube apps) did not constitute a "playback queue" because they "merely provide the means to *process* the lists for playback. In short, the cloud queue runs the show." *Id.*, 10.

305. In view of Google and Dr. Bhattacharjee's representations to the Court, I disagree with Dr. Bhattacharjee opinion that "[t]he Court's order supports [his] opinion that the asserted claims of the '033 patent are invalid …." Bhatta. Op. Report, ¶263.

Report, ¶299. I disagree with Dr. Bhattacharjee. As explained above, in standalone mode, the remote controls do not playback any videos from anywhere but their ***local*** playback queue. In order for any recommended videos to be played back by a remote control, a user would have to add such videos to the local playback queue on the remote control. This is evidenced by the "+" icon that appears next to a recommended video. *See, e.g.*, Bhatta. Op. Report, ¶167. In the YTR System, playback queues are not auto-populated with recommended videos. Accordingly, in standalone mode, a remote control only plays back a recommended video if that video has been added to its local playback queue, and in that instance, only plays back the recommended video from its local playback queue.

329. Dr. Bhattacharjee also opines that "the prior art YTR application satisfies this limitation when playing a 'party queue' in standalone mode." Bhatta. Op. Report, ¶300. I disagree with Dr. Bhattacharjee. As explained above, in standalone party mode, the remote controls do not playback any videos from anywhere but their local playback queue. Each remote control gets a copy of the entire party playlist, which it loads into a local playback queue. From there, each remote control plays back the media items that are stored in its local playback queue, not a remote playback queue.

330. In other words, in the YTR System's standalone party and non-party modes, the playback queues locally stored on the remote controls: (i) contain the list of media items that are used for playback by the remote controls; (ii) contain the entire list of media items selected for playback; (iii) are not being used merely to process the list of media items for playback; and (iv) are the queues that "run[] the show." Consequently, the remote controls in the YTR System have ***local*** playback queues, not ***remote*** playback queues.

331. It is therefore my opinion that Dr. Bhattacharjee failed to establish that the YTR System invalidates limitation 1.4.

"+" icons in the remote control's GUI) and then effectively transfer that list of videos via the Lounge Server to the local playback queues of the connected Leanback Screens. Accordingly, in remote-control non-party mode, a Leanback Screen only plays back a recommended video if that video has been added to its local playback queue, and in that instance, only plays back the recommended video from its local playback queue.

359. Dr. Bhattacharjee also opines that "limitation [1.7] is met when a user is in party mode and transfers playback of a party queue." Bhatta. Op. Report, ¶327. I disagree with Dr. Bhattacharjee. As explained above, in remote-control party mode, the Leanback Screens do not playback any videos from anywhere but their local playback queue. Each Leanback Screen gets a copy of the entire party playlist, which it loads into a local playback queue. From there, each Leanback Screen plays back the media items that are stored in its local playback queue, not a remote playback queue.

360. In other words, in the YTR System's remote-control party and non-party modes, the playback queues locally stored on the Leanback Screens: (i) contain the list of media items that are used for playback by the Leanback Screens; (ii) contain the entire list of media items selected for playback; (iii) are not being used merely to process the list of media items for playback; and (iv) are the queues that "run[] the show." Consequently, the Leanback Screens in the YTR System have ***local*** playback queues, not ***remote*** playback queues.

361. It is therefore my opinion that Dr. Bhattacharjee failed to establish that the YTR System invalidates limitation 1.7.

e. **[1.8]** "*detecting an indication that playback responsibility for the remote playback queue has been successfully transferred from the computing device to the at least one given playback device*"

362. As I explained above, Dr. Bhattacharjee failed to establish that the YTR System invalidates limitations 1.4 and 1.7. For instance, Dr. Bhattacharjee failed to establish that either the remote controls or the Leanback Screens in the YTR System use a "remote playback queue" for playback. It follows then that Dr. Bhattacharjee failed to establish that the YTR System invalidates limitation 1.8, which requires a "successful transfer[]" of "playback responsibility for the remote playback queue" from the "computing device" to the " playback device."

available at the time of the alleged first infringement [September 15, 2020] (and are still available today)." Bhatta. Op. Report, ¶751. I disagree for the reasons I explained in my Opening Report (Schmidt Op. Report, ¶¶473-527), which I incorporate by reference herein, and the reasons I explain below.

A. Alternative #1 – "Playback Of A Local Queue On The Sender Device"

1003. Dr. Bhattacharjee states that what he labels "Alternative #1" involves "play[ing] back a local queue on the sender device," such that "the sender device would maintain the queue locally and the sender device would be configured to playback the local queue." Bhatta. Op. Report, ¶754. In this regard, Dr. Bhattacharjee opines that "Alternative #1" would not satisfy limitation 1.4 because "the sender devices would not be configured to playback 'a remote playback queue provided by a cloud-based computing system associated with a cloud-based media service." *Id.* I disagree with Dr. Bhattacharjee for many of the same reasons I discussed in my Opening Report in connection with Google's alleged "non-infringing alternative" #2 as well as the reasons set forth below. *See* Schmidt Op. Report, ¶¶478-95.

1004. **First**, I already provided some examples as to why Google has not provided sufficient details as to this alleged alternative in my Opening Report. *Id.*, ¶480. I understand that Dr. Bhattacharjee has not provided any more details than what Google previously provided with respect to this alleged alternative. As a result, I do not have enough information to fully evaluate whether this alleged alternative would have been non-infringing, available, technically feasible, or commercially acceptable. Nevertheless, I have made my best effort to respond to this alleged alternative based on my current understanding of the limited information provided by Google. I expressly reserve the right to supplement my opinions regarding this alleged alternative if and when Google provides sufficient details.

1005. **Second**, Dr. Bhattacharjee does not provide sufficient evidence to support his opinion that this alleged alternative "would have been an available option for Google to implement at the time of the alleged first infringement and would take little engineering time and cost to implement." Bhatta. Op. Report, ¶755. Specifically, Dr. Bhattacharjee opines that this alleged alternative "would take little engineering time and cost to implement" because "the source code for the accused YouTube applications already includes a file DefaultPlaybackQueue.java that implements a QUEUE_LIST and AUTONAV_LIST that is stored locally on the device from which playback occurs." *Id.* However, as I previously explained in my Opening Report, just because a sender device might ***maintain*** a local copy of a queue does not necessarily mean that there is not also "a remote playback queue ***provided by*** a cloud-based computing system associated with a cloud-based media service," and that "remote playback queue" "runs the show." It seems to me that Dr. Bhattacharjee is advancing inconsistent positions that there can only be one "playback queue" in a system for purposes of infringement but there can be multiple "playback queues" (both a "remote playback queue" and a "local playback queue") for the purposes of invalidity, despite his representations to the Court that say otherwise.

1006. Dr. Bhattacharjee also relies on his experience and a call with Mr. Nicholson to opine that he "estimate[s] that the modification to all the relevant accused products after they were launched would have required at three Level-4 to Level-5 software engineers working for approximately less than three months to implement this alternative." Bhatta. Op. Report, ¶756. I do not find this evidence sufficient because Dr. Bhattacharjee has not identified any specific experience with projects of this magnitude and Mr. Nicholson's lack of involvement with the infringing Hub devices and the YouTube apps other than the YouTube Music app. *See, e.g.,* Nicholson Dep. Tr., 14:13-14 (Mr. Nicholson testifying that he works on the YouTube Music team), 23:12-14 (Q Do you work on anything involving Dragon Glass? A No."), 24:7-9 (Mr. Nicholson testifying that he never worked on the YouTube Main team).

1007. **<u>Third</u>**, I disagree with Dr. Bhattacharjee opinion that "end users would have found Alternative #1 to be an acceptable alternative." Bhatta. Op. Report, ¶755. As explained in my Opening Report, it is my opinion that this alleged alternative would not have been technically feasible (or not technically feasible without introducing inefficiencies) and would not have been commercially acceptable. *See* Schmidt Op. Report, ¶¶486-95.

1008. I therefore disagree with Dr. Bhattacharjee that what he labels "Alternative #1" would have been an available, acceptable non-infringing alternative at the time of Google's first infringement.

Douglas C. Schmidt

Dated: January 13, 2023

DOUGLAS C. SCHMIDT